SpringBeansFromFlowScript

Spring Framework from FlowScript

This is a short introduction on how you can access Spring Framework beans from your flowscript. See http://www.springframework.org for more information about Spring, and the SpringPetstore example.

Why use Spring?

Spring lets you configure and instantiate Java objects using an *application context* XML file. This can be a simple list of your *Singleton* Java classes for which there should only ever be one instance created throughout the entire application. An example of a Singleton would be a class which interacts with a database - each instance of that class would have its own database connection, so having more than one would be a waste of resources. The Spring Framework allows you to create exactly one instance of each singleton, passing it any parameters needed; which could be a simple text value, a list of values, or a reference to another bean defined in the application context. All you have to do to let Spring inject a value is to add a setter and getter method for the properties you want set.

Example applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN" "http://www.springframework.org/dtd/spring-beans.dtd">

<beans>
  <!-- Define your data source -->
  <bean id="myDataSource" class="org.apache.commons.dbcp.BasicDataSource" destroy-method="close">
    <property name="driverClassName">
      <value>org.hsqldb.jdbcDriver</value>
    </property>
    <property name="url">
      <value>jdbc:hsqldb:hsql://localhost:9002</value>
    </property>
    <property name="username">
      <value>sa</value>
    </property>
    <property name="password">
      <value></value>
    </property>
  </bean>

  <!-- Pass a reference to the data source to a Data Access Object -->
  <bean id="userDAO" class="org.myapp.UserDAOImpl">
    <property name="dataSource">
      <ref bean="myDataSource"/>
    </property>
  </bean>

  <!-- Pass a reference to the data access object to your application -->
  <bean id="myApp" class="org.myapp.MyAppImpl">
    <property name="userDAO">
      <ref bean="userDAO"/>
    </property>

  </bean>
</beans>
```

Accessing your application bean from Flowscript

Since Cocoon currently uses Avalon, you must create a wrapper object which it initializes on startup. You can then access this wrapper from your flowscript and retrieve a reference to your application bean, which has been injected with properties by Spring.

```
public class MyAppWrapper implements Initializable {
    private MyAppService service;

    public MyAppService getService() {
        return service;
    }

    public synchronized void initialize() throws Exception {
        BeanFactoryLocator bfl = org.springframework.beans.factory.access.SingletonBeanFactoryLocator
                .getInstance("net/sanctuarytechnology/timesheets/spring/beanRefFactory.xml");

        BeanFactory beanFactory = bfl.useBeanFactory("org.myapp.spring.appCtx").getFactory();

        service = (MyAppService ) beanFactory.getBean("
        LocalSessionFactoryBean lsfb = (LocalSessionFactoryBean)
        service.initializeDatabase(lsfb);

    }
}
```

then in your flowscript

```
importClass(Packages.org.myapp.MyAppWrapper);

var myApp = cocoon.createObject(MyAppWrapper).service;
```